EAX' Security Explained

Edward J. Beroset
Elster Solutions, LLC.
edward.j.beroset@us.elster.com

June 4, 2010

1 Introduction

The electric meter communications protocol ANSI C12.22/IEEE 1703, relies on a provably secure, fast and lightweight security mechanism called EAX'[1]. The purpose of the brief and informal paper is to explain how this mechanism works in a way that is intelligible to both security researchers and protocol implementers.

2 Security Primitives

The ANSI C12.22/IEEE 1703 security mechanism relies on NIST-recommended ciphers and modes. Namely, it uses AES-128[2] as the symmetric block cipher in the CBC and CTR modes[3] and it uses a standard Message Authentication Mode, CMAC[4].

The entire EAX' protocol, which is derived from EAX[5] is shown in block-diagram in Figure 1. The diagram shows how the inputs, which include the header data (which remains unencrypted) and also the payload data (which is encrypted) and the encryption key.

To understand this diagram completely, it is necessary to describe the primitives that are included within the diagram. These are *pad*, *AES*, *dbl* and the simple bitwise operations *XOR*, *AND*, *NOT*, *first 32-bits*. It is also necessary to understand the meaning of the subdiagram that defines the generation of D and Q from the key K.

2.1 Deriving Key Dependent Constants

Constants D and Q are derived from the key K as follows. First a 128-bit block of zeroes is encrypted using the secret key K. The result of that operation is then input to the *dbl* operation and the output of that is D. Q is then derived from D by another application of the *dbl* operation. In psuedocode,

```
D = dbl(AES(K, 0))
Q = dbl(D)
```

These operations are described in more detail in the next sections.

2.2 The *dbl* operation

The *dbl* operation is part of the CMAC mechanism. If the most significant bit of the input is 0, the output is the input left shifted by one bit (with a 0 carry-in). If the most significant bit of the input is 1, the output is the left-shifted version of the input XORed with 0x0087. Both input and output are 128-bit values. In pseudocode, this can be expressed as follows:

```
dbl(X) {
    if (msb(X) == 0) then
        return X<<1
    else
        return (X<<1) XOR 0x0087
}
```

2.3 The *AES* operation

In this document, we consider the *AES* operation to be a primitive which is the application of the AES-128 algorithm, as defined in FIPS-197[2]. The inputs to the operation are the key K and the input data. In pseudocode, this is

```
AES(K, data)
```

2.4 The *pad* operation

The pad operation is also part of the CMAC mechanism and is responsible for padding the input blocks to an integral multiple of full 128-bit blocks. The input is at least one bit. If the input is already an integral multiple of 128-bit blocks long, the output is the input with D XORed onto the end (last block) of the input. If the input is not an integral multiple of 128-bit blocks long, the output is padded with a 1 bit and as many 0 bits as needed to make the size an integral multiple of 128-bit blocks and then XORed with Q. In psuedocode, this can be expressed as:

```
pad(data, D, Q) {
    if (length of data is an integral multiple of 128-bit blocks) then
        return data XOR D
    else
        return (data appended with 1 and 0's to make 128-bit multiple) XOR Q
}
```

2.5 Simple Bitwise Operations

The remaining operations, namely *XOR*, *AND*, *NOT*, *first 32-bits* all have conventional meanings. So *XOR* is a bitwise exclusive-OR of one 128-bit block with another to create an output. The *AND* operation is similarly a bitwise AND of one 128-block with another to create an output. The *NOT* operation is takes a single input and simply does a bitwise inversion of a single 128-bit block to create an output. The *first 32-bits* operation simply extracts the most significant 32 bits of the input 128-bit block. It is only used once to create the value by which both the encrypted data and the unencrypted header data may be validated.

3 Analysis

In the case of EAX', one important attribute is that the nonce N used to create the initial value of the counter for the CTR mode must be unique over the lifetime of the key. That is, if the same key and the same nonce were used for data encryption, this may provide a vector for attack. However, an analysis of the algorithm, as shown in Figure 1, reveals that the nonce N is derived from the incoming message header and the secret key K. Since the nonce N must be unique if the key K does not vary, this means that the header must be unique over the lifetime of the key. In fact, C12.22 contains three elements which are used to assure the uniqueness of a header. They are an *iv-element* which is expressly for this purpose and is tied to the time of day, and also

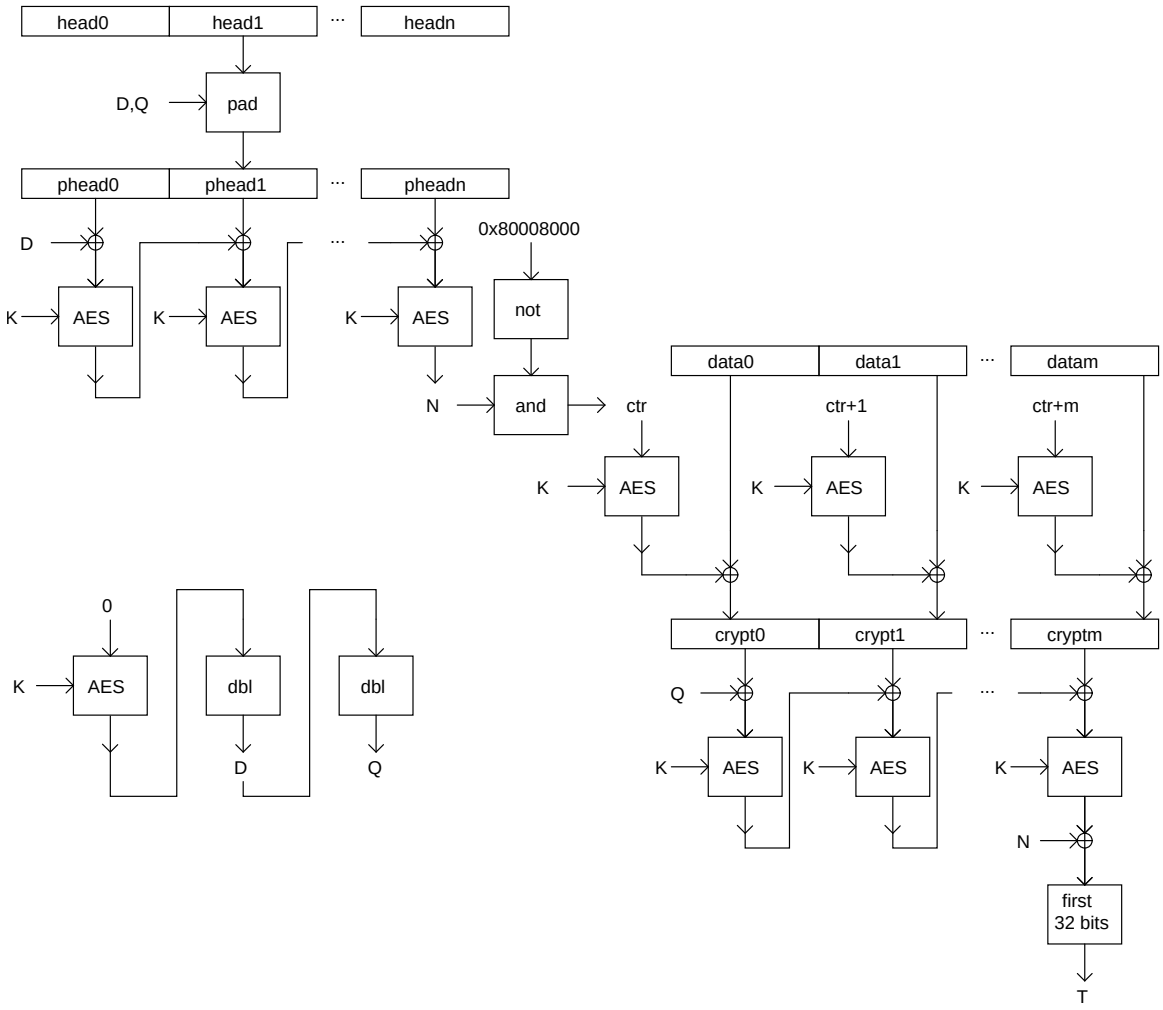

Figure 1: EAX' encryption algorithm

two other values, which are known as the *called-AP-invocation-id* and the *calling-AP-invocation-id*. These latter two are both arbitrary 32-bit unsigned integers which allow for requests and responses to be matched in the protocol. The combination of these elements allows every header to be unique over the lifetime of the key.

References

[1] "Protocol specification for interfacing to data communication networks," standard, American National Standards Institute, 2008.

[2] "Fips 197, advanced encryption standard," Federal Information Processing Standard, National Institute of Standards and Technology, 2001. [Online]. Available: http://www.csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[3] "Sp 800-38a, recommendation for block modes of operation; methods and techniques," Special Publication, National Institute of Standards and Technology, 2001. [Online]. Available: http://www.csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

[4] "Sp 800-38b, recommendation for block cipher modes of operation; the cmac mode for authentication," Special Publication, National Institute of Standards and Technology, 2005. [Online]. Available: http://www.csrc.nist.gov/publications/nistpubs/800-38b/sp800-38b.pdf

[5] M. Bellare, P. Rogaway, and D. Wagner, "A convential authenticated-encryption mode," PDF document, 2005. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/eax/eax-spec.pdf